Predictive Maintenance using the AI4I 2020 Dataset

Trevor King

2024-05-14

Contents

Introduction

The dataset was downloaded from Kaggle and is modeled after an existing milling machine. The dataset has 10000 rows with 14 columns. The 14 columns contain information about the product ID and type as well as temperature, tourque, rpm, tool wear, and 6 columns for failure data.

ai4i2020.csv

The Fields are as follows:

1. UID
2. Product ID Unique values (Prefixed with L - Low, M - Medium, and H - High)
3. Product Type product type L, M or H from 2 above
4. air temperature [K]
5. Process temperature [K]
6. Rotational Speed [RPM]
7. Torque [Nm]
8. Tool wear [min]
9. Target
10. Machine failure

Methods/Analysis

Exploratory Data Analysis

A preview of the table structure.

```r
# Read in the CSV file
FN <- "C:/Users/kingt/Documents/CapstonePredictiveMaintenance/data/predictive_maintenance.csv"

x<-spec_csv(FN,col_types = cols())

read_csv(FN)%>%apply(., 2, function(x) {length(unique(x))})%>%kable(col.names = c('Length'))
```

```
## Parsed with column specification:
## cols(
##   UDI = col_double(),
##   `Product ID` = col_character(),
##   Type = col_character(),
##   `Air temperature [K]` = col_double(),
##   `Process temperature [K]` = col_double(),
##   `Rotational speed [rpm]` = col_double(),
##   `Torque [Nm]` = col_double(),
##   `Tool wear [min]` = col_double(),
##   Target = col_double(),
##   `Failure Type` = col_character()
## )
```

	Length
UDI	10000
Product ID	10000
Type	3
Air temperature [K]	93
Process temperature [K]	82
Rotational speed [rpm]	941
Torque [Nm]	577
Tool wear [min]	246
Target	2
Failure Type	6

Note that the Type only has 3 distinct values which makes it a good candidate for factorization.

And the last five columns only have two values, 0 and 1, we can bring these in as logical

```r
# Import the data with defined field types
maintlog<-read_csv(FN,col_types = 'icfddiddlf')

# Display the 1st 10 rows
maintlog%>%head()%>%kable()
```

UDI	Product ID	Type	Air temperature [K]	Process temperature [K]	Rotational speed [rpm]	Torque [Nm]	Too
1	M14860	M	298.1	308.6	1551	42.8	
2	L47181	L	298.2	308.7	1408	46.3	
3	L47182	L	298.1	308.5	1498	49.4	
4	L47183	L	298.2	308.6	1433	39.5	
5	L47184	L	298.2	308.7	1408	40.0	
6	M14865	M	298.1	308.6	1425	41.9	

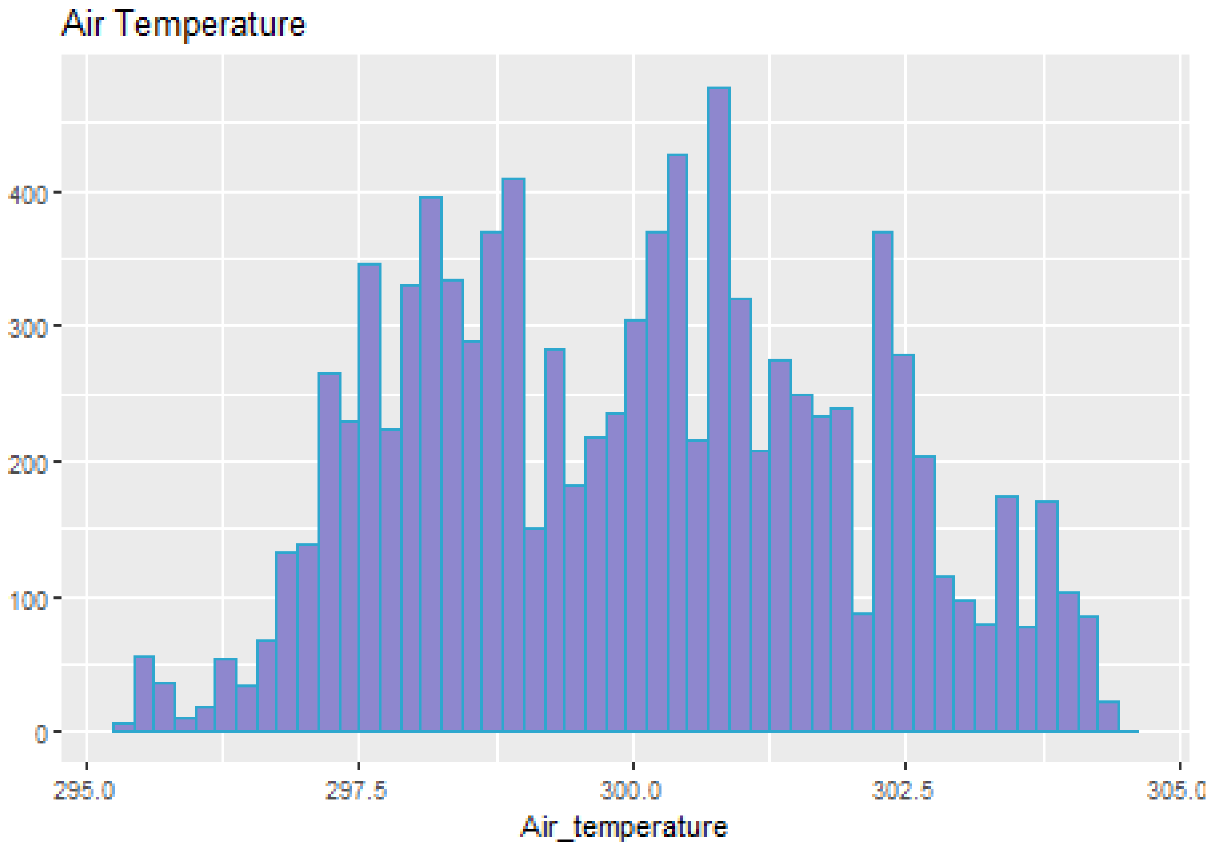
```
# Remove the squre brackets from the vars
names(maintlog)<-names(maintlog)%>%sub("\\s\\[.{1}.*\\]$", "", .)

# Replace the spaces with '_' in the column names (Easier to work with)
names(maintlog)<-names(maintlog)%>%gsub("\\s", "_", .)

value_features<-names(maintlog)[4:8]

# Create a pair plot of
vf<-maintlog[,4:10]
# vf%>%pairs(col = 'blue', #modify color
#       main = 'Features vs. Machine Failure')
```

Numeric Variables

Quantitative variables

Air Temperature

```
vf%>%qplot(Air_temperature, geom ="histogram", data = .,
colour = I("#2EA7CE"), fill = I("#8E87CE"),
main = "Air Temperature", bins=50)
```

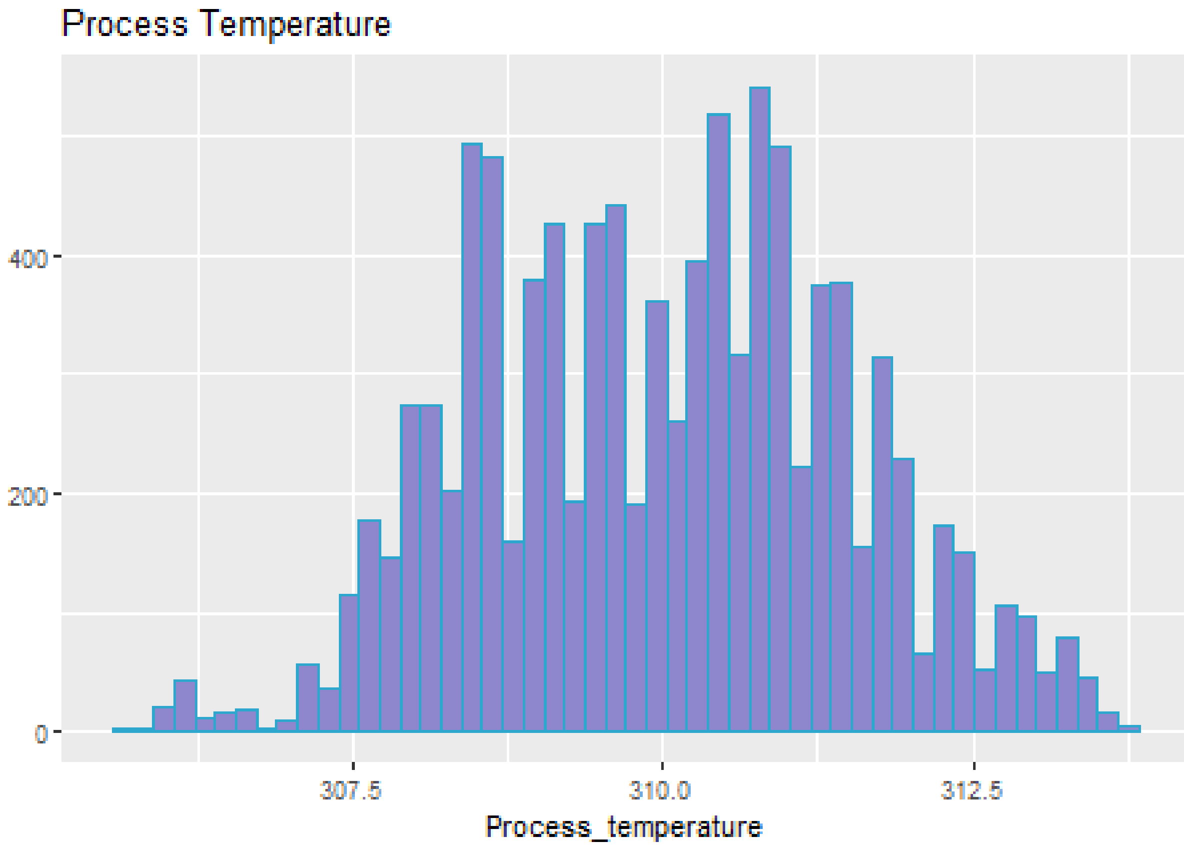



```
summary(vf$Air_temperature)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   295.3   298.3   300.1   300.0   301.5   304.5
```

Process Temperature

```
vf%>%qplot(Process_temperature, geom ="histogram", data = .,
colour = I("#2EA7CE"), fill = I("#8E87CE"),
main = "Process Temperature", bins=50)
```

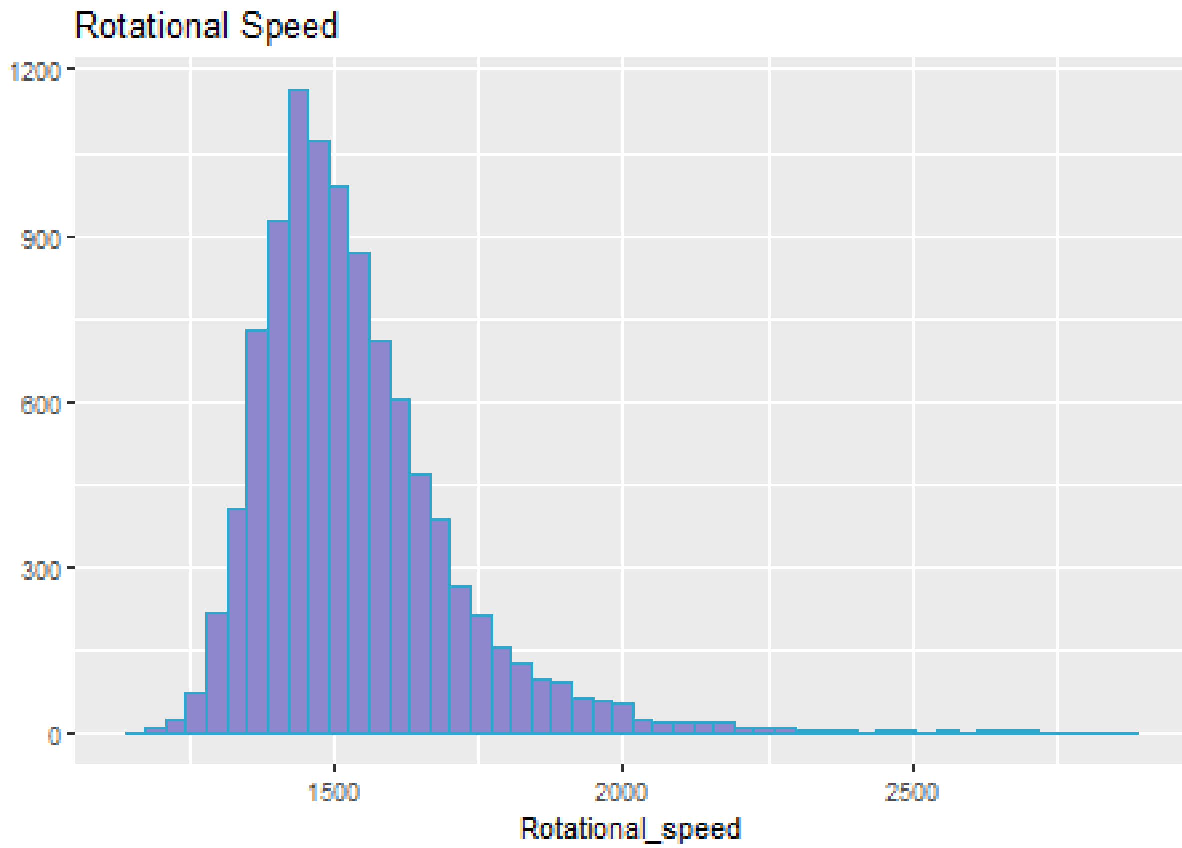



```
vf$Process_temperature%>%summary()
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   305.7   308.8   310.1   310.0   311.1   313.8
```

Rotational Speed

```
vf%>%qplot(Rotational_speed, geom ="histogram", data = .,
colour = I("#2EA7CE"), fill = I("#8E87CE"),
main = "Rotational Speed", bins=50)
```

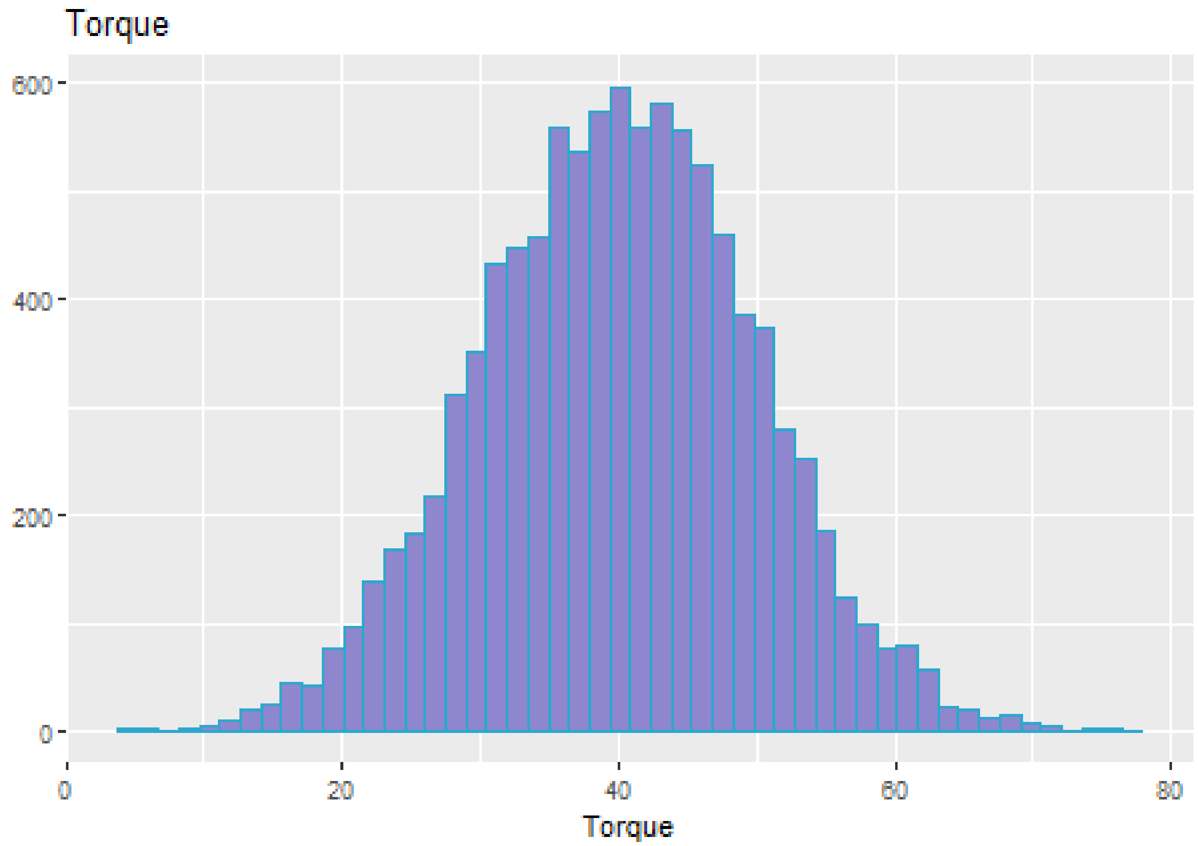



```
vf$Rotational_speed%>%summary()
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    1168    1423    1503    1539    1612    2886
```

Torque

```
vf%>%qplot(Torque, geom ="histogram", data = .,
colour = I("#2EA7CE"), fill = I("#8E87CE"),
main = "Torque", bins=50)
```

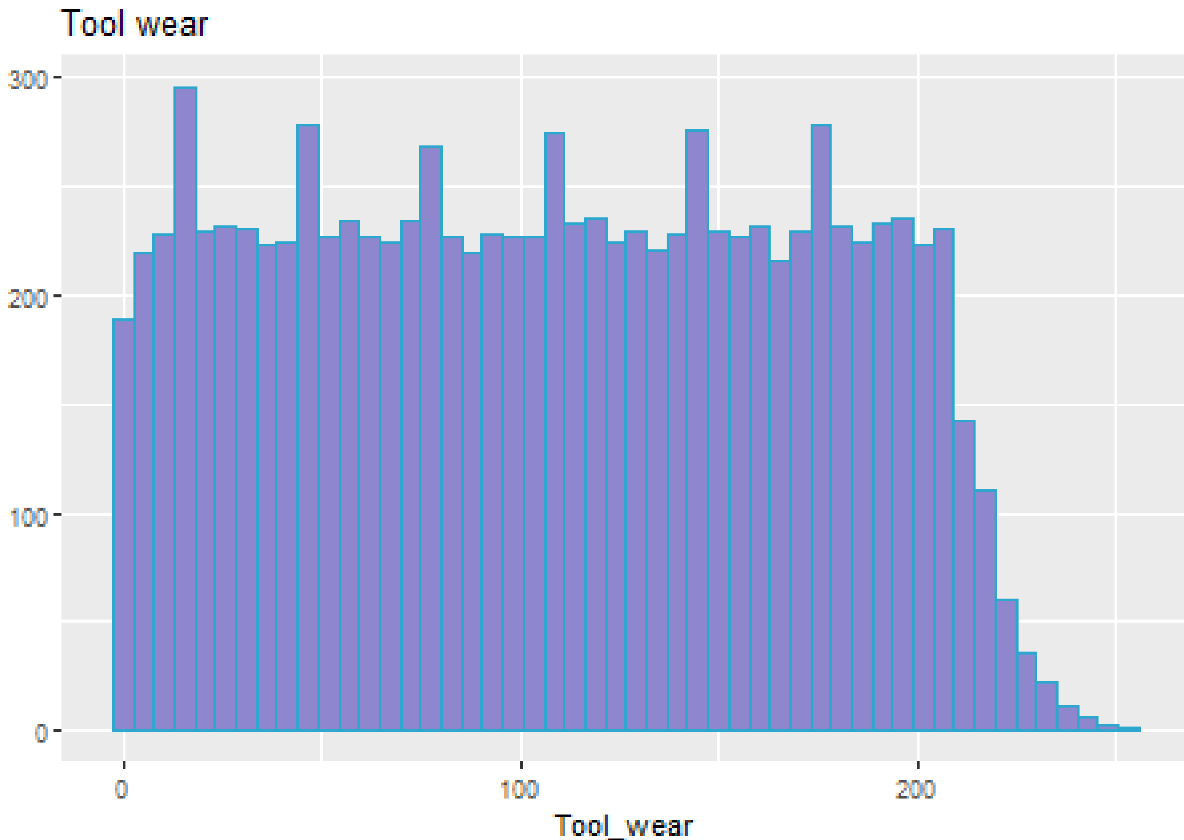



```
vf$Torque%>%summary()
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    3.80   33.20   40.10   39.99   46.80   76.60
```

Tool wear

```
vf%>%qplot(Tool_wear, geom ="histogram", data = .,
colour = I("#2EA7CE"), fill = I("#8E87CE"),
main = "Tool wear", bins=50)
```




```
vf$Tool_wear%>%summary()
```

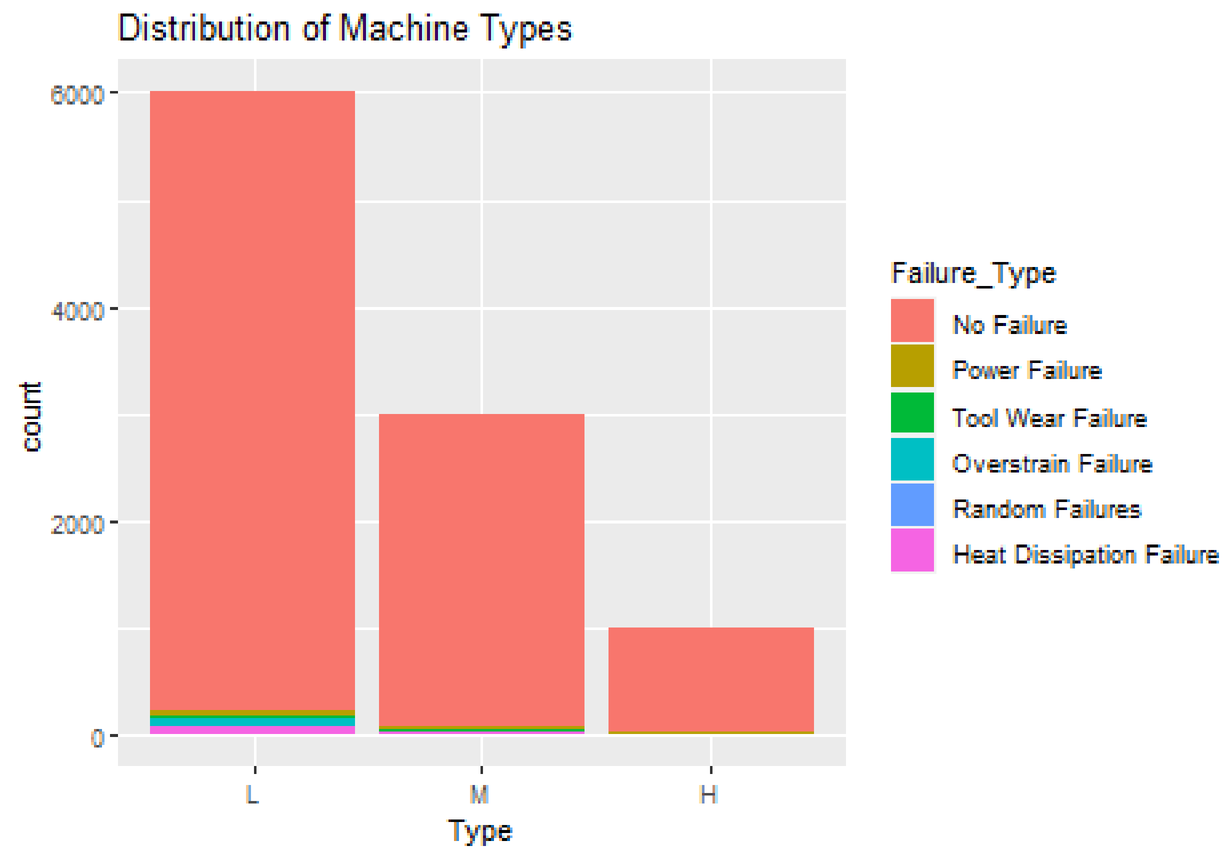
```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##       0      53     108     108     162     253
```

```
maintlog$Type <- factor(maintlog$Type, levels=c('L', 'M', 'H'))

maintlog$Type%>%table()%>%kable()
```

.	Freq
L	6000
M	2997
H	1003

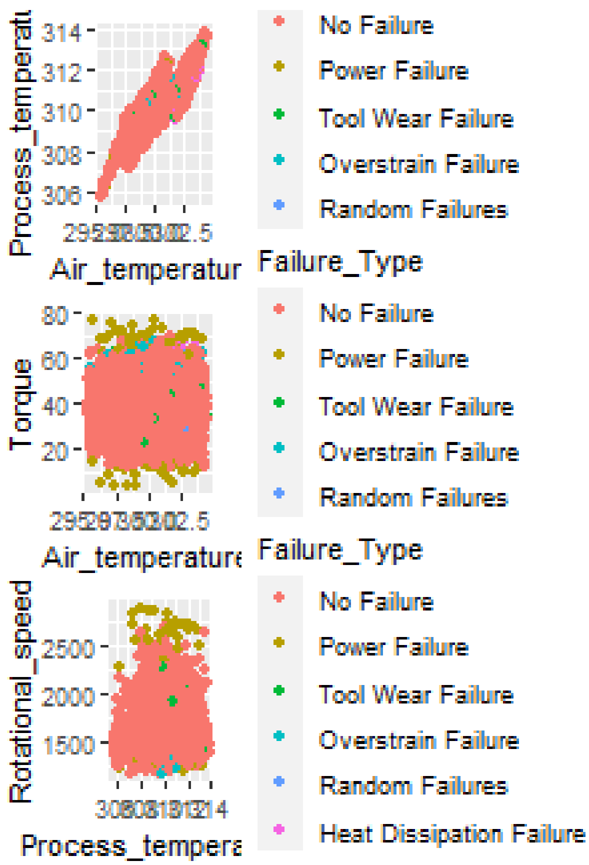
```
maintlog%>%ggplot(aes(Type, fill = Failure_Type)) +
  geom_histogram(stat="count") + ggtitle("Distribution of Machine Types")
```

```
## Warning: Ignoring unknown parameters: binwidth, bins, pad
```

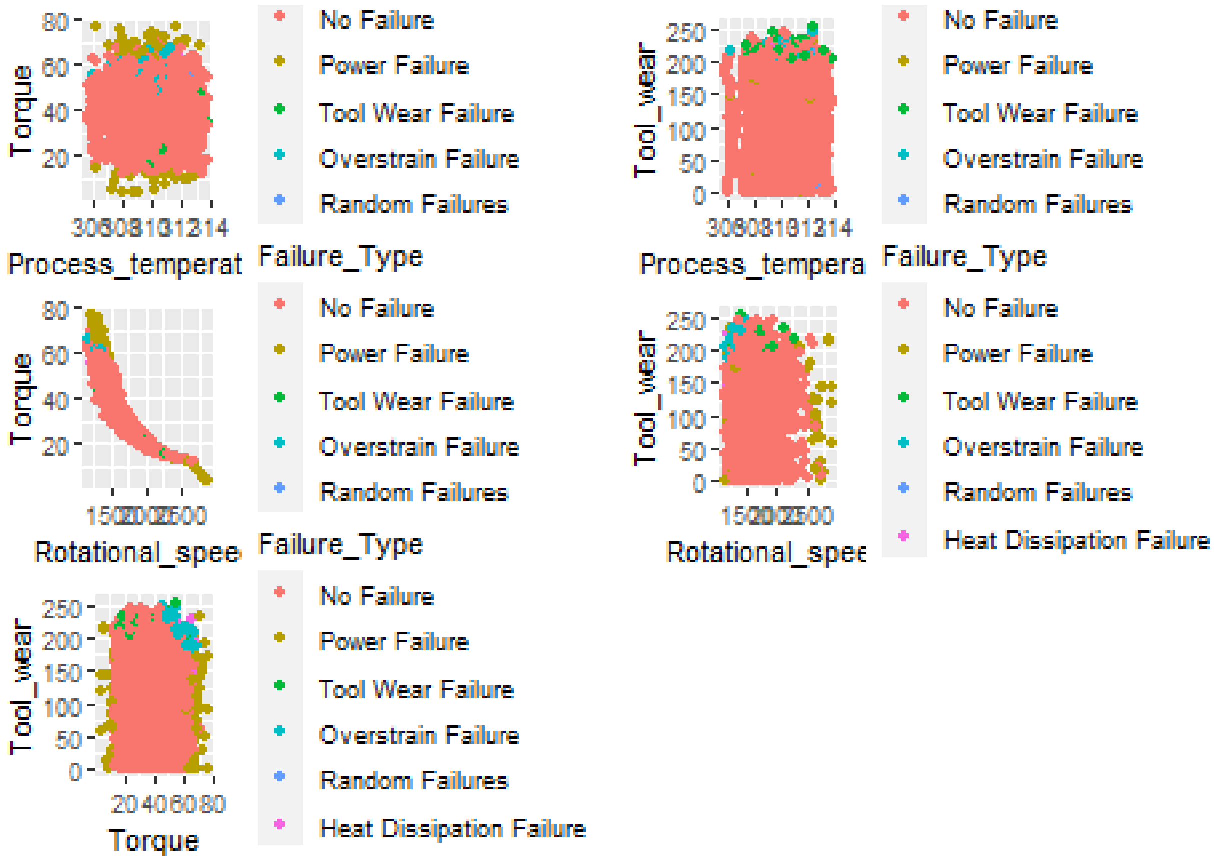



```r
# Air Temp vs. Proccess Temp
p1 <- ggplot(vf)+geom_point(aes(x = Air_temperature, y = Process_temperature,colour = Failure_Type), na

# Air Temp vs. Rotation Speed
p2 <- ggplot(vf)+geom_point(aes(x = Air_temperature, y = Rotational_speed,colour = Failure_Type), na.rm

# Air Temp vs. Torque
p3 <- ggplot(vf)+geom_point(aes(x = Air_temperature, y = Torque,colour = Failure_Type), na.rm = TRUE)

# Air Temp vs. Tool wear
p4 <- ggplot(vf)+geom_point(aes(x = Air_temperature, y = Tool_wear,colour = Failure_Type), na.rm = TRUE

# Process temp vs. Rotation Speed
p5 <- ggplot(vf)+geom_point(aes(x = Process_temperature, y = Rotational_speed,colour = Failure_Type), na

# Process temp vs. Torque
p6 <- ggplot(vf)+geom_point(aes(x = Process_temperature, y = Torque,colour = Failure_Type), na.rm = TRUE

# Process temp vs. Tool wear
p7 <- ggplot(vf)+geom_point(aes(x = Process_temperature, y = Tool_wear,colour = Failure_Type), na.rm =

# Rotation Speed vs. Torque
p8 <- ggplot(vf)+geom_point(aes(x = Rotational_speed, y = Torque,colour = Failure_Type), na.rm = TRUE)

# Rotation Speed vs. Tool wear
p9 <- ggplot(vf)+geom_point(aes(x = Rotational_speed, y = Tool_wear,colour = Failure_Type), na.rm = TRUE
```

```r
# Torque vs. Tool wear
p10 <- ggplot(vf)+geom_point(aes(x = Torque, y = Tool_wear,colour = Failure_Type), na.rm = TRUE)


# observe how these features interact under different failure conditions.
gridExtra::grid.arrange(p1,p2,p3,p4,p5)
```




```r
gridExtra::grid.arrange(p6,p7,p8,p9,p10)
```

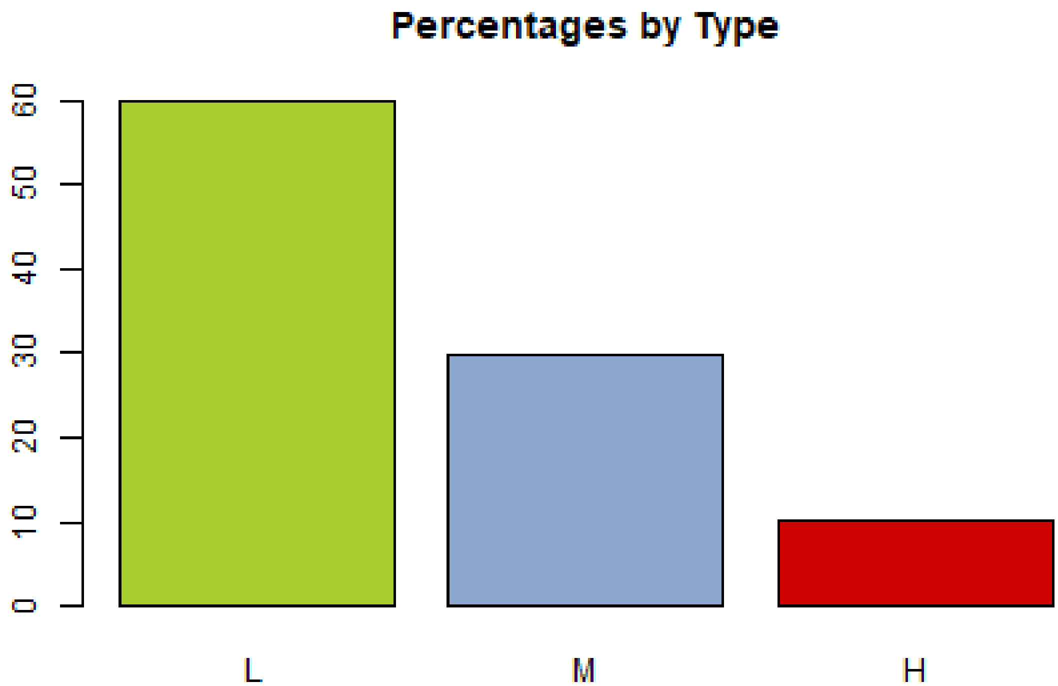


Creating the training, testing and validation sets

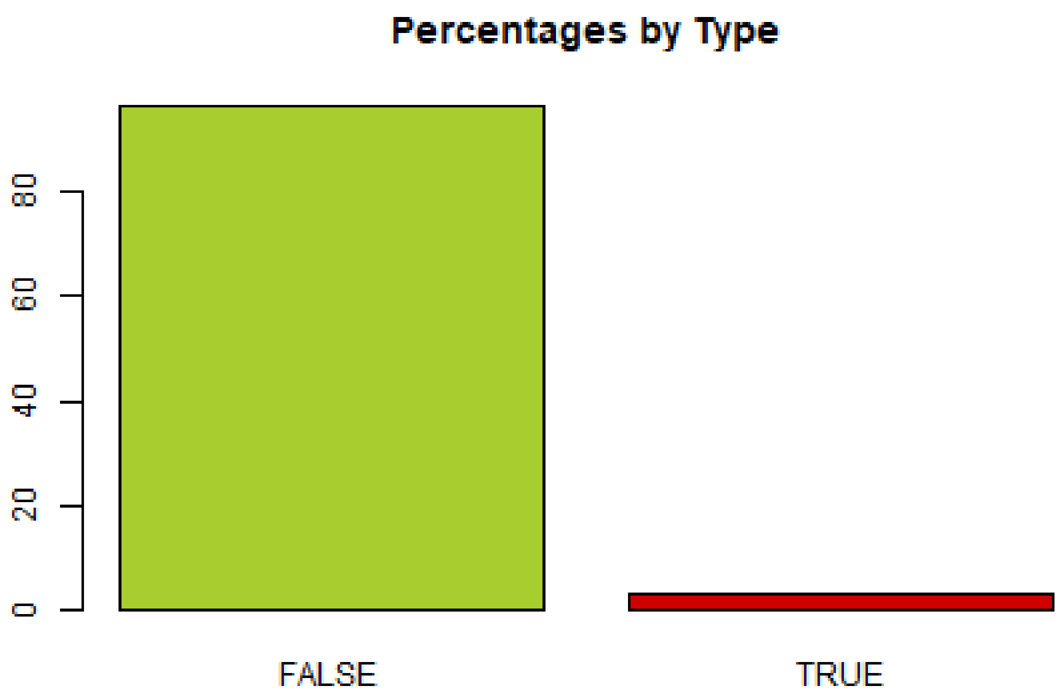
```r
# Need to check the R version since the syntax is different for R version>3.6
# Set the random seed value
if(base::getRversion()>'3.6'){ set.seed(1, sample.kind="Rounding") }else{set.seed(1) }

# Partition the data by creating an index where 10% is for the hold out data and
# 90% for the remain
test_index <- createDataPartition(y = maintlog$UDI, times = 1, p = 0.1, list = FALSE)
train_set <- maintlog[-test_index,]
test_set <- maintlog[test_index,]
```

```
## Warning: The `i` argument of ``[`()` can't be a matrix as of tibble 3.0.0.
## Convert to a vector.
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_warnings()` to see where this warning was generated.
```

```r
plt<-table(train_set$Type)/length(train_set$Type)*100
barplot(plt, col = c(I("#a7ce2e"),I("#8EA7CE"),I("#CE0000")),main = "Percentages by Type")
```

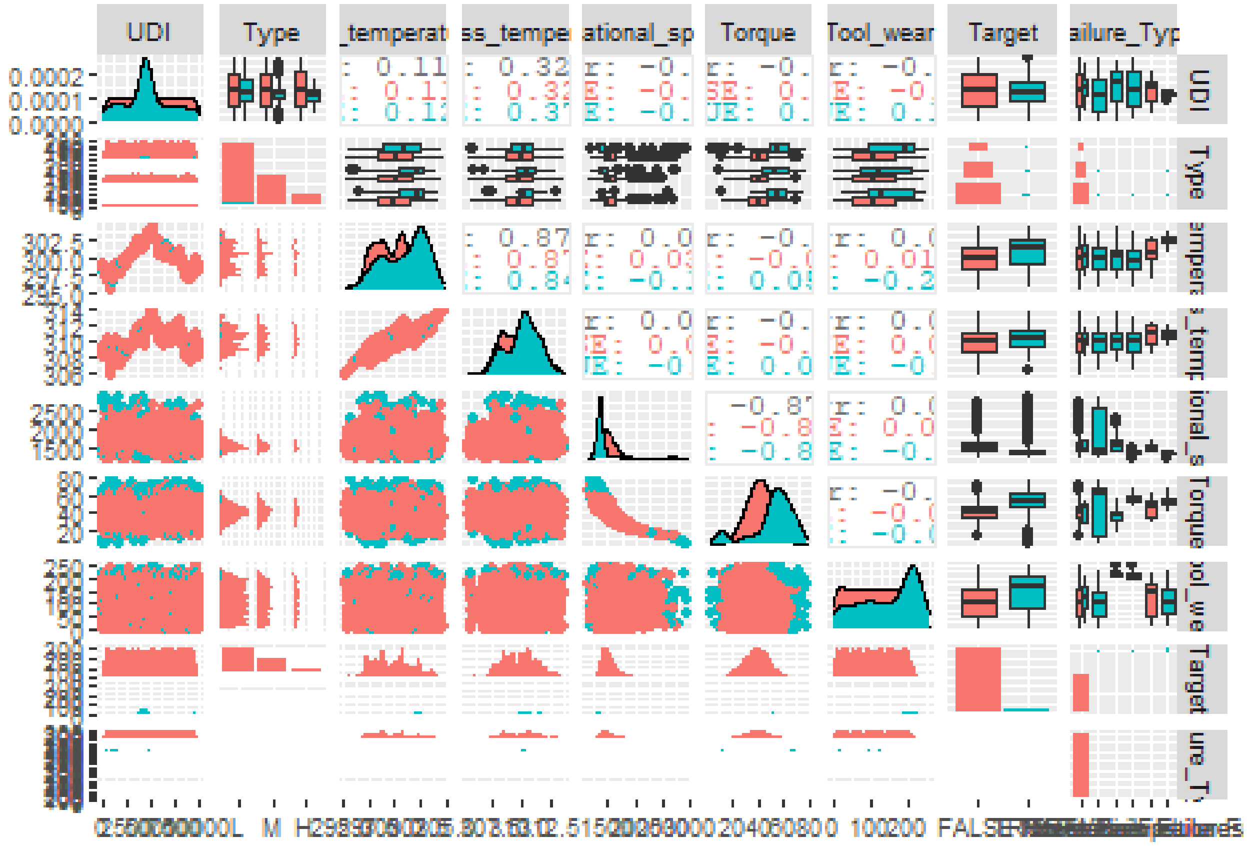



```
plt<-table(train_set$Target)/length(train_set$Target)*100
barplot(plt, col = c(I("#a7ce2e"),I("#CE0000")),main = "Percentages by Type")
```

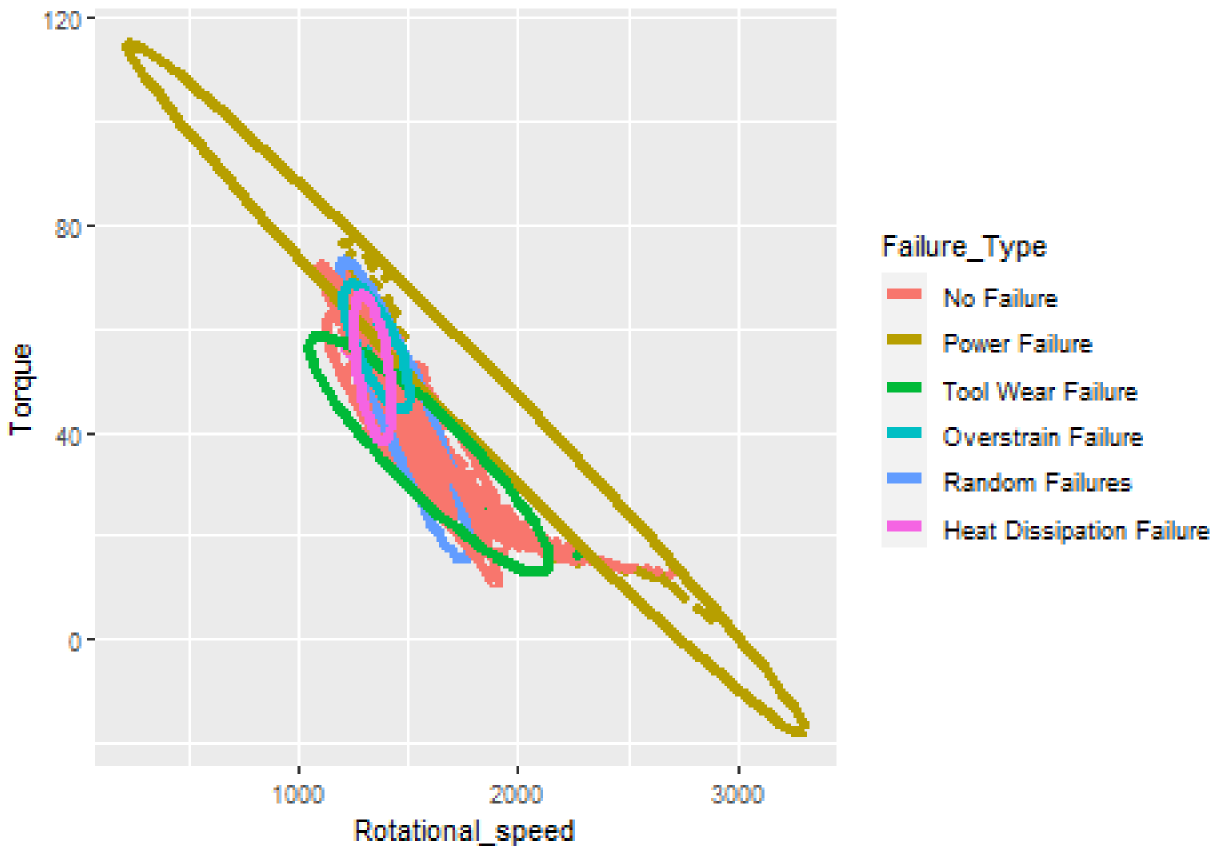



```
train_qvar <- train_set[, - 2]
Target<- as.factor(train_set$Target)
ggpairs(train_qvar, aes(colour=Target))
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```




```r
train_set[,-1]%>%
  ggplot(aes(Rotational_speed, Torque, fill = Target, color=Failure_Type)) +
  geom_point(show.legend = FALSE) +
  stat_ellipse(type="norm", lwd = 1.5)
```




```
# Logistic regression

# train_qda <- train(Failure_Type ~ ., method = "qda", data = train_set[,-1])
```

Results

Conclusion

references

S. Matzka, "Explainable Artificial Intelligence for Predictive Maintenance Applications," 2020 Third International Conference on Artificial Intelligence for Industries (AI4I), 2020, pp. 69-74, doi: 10.1109/AI4I49448.2020.00023.